2022 8th International Conference on Control, Decision and Information Technologies
CoDIT'22 | Istanbul, Turkey / May 17-20, 2022

Experimental Evaluation of the Success of Peg-in-Hole Tasks Learned from Demonstration

Serdar Hakan Argüz[1] , Şeniz Ertuğrul[2] and Kerem Altun[3]

2022 8th International Conference on Control, Decision and Information Technologies (CoDIT) | 978-1-6654-9607-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/CODIT55151.2022.9804111

Abstract— **Industrial robots are traditionally programmed by hard-coding the desired motion into them. That approach, however, costs significant time and effort and shows little to no promise in transferring human skills to robots. Programming by demonstration (PbD) is an alternative approach that allows robots to learn tasks from demonstrations. Because of its several advantages over the traditional method, PbD is particularly suited for tasks encountered in assembly operations, the most typical of which is the peg-in-hole task. A successful PbD implementation for a peg-in-hole task requires that the peg should still be inserted into the hole even under situations that are not encountered during the demonstrations. Previous research in the field shows that the success rate of a peg-in-hole task under such cases varies greatly. In this study, we use a UR5 manipulator to experimentally investigate how the success rate of a peg-in-hole task changes with respect to the novelty of the task, quantified in terms of the distance of the hole to its original position. It is found that the success ratio decreases as the novelty of the task increases. To increase the performance, the use of strategies that alter the robot's motion dynamically in the run time is suggested for future work.**

I. INTRODUCTION

Robots, by their very definition, are reprogrammable to perform a variety of tasks. Traditionally, this reprogramming is done by the hard-coding of the desired motion into the robot. However, the time and effort associated with this approach have led the field to consider some alternatives. Programming by Demonstration (PbD) is one such alternative that attempts to teach the task to the robot by providing demonstrations of it.

The advantages of PbD over the traditional programming can be listed as follows [1] [2] [3] [4] [5]:

- PbD takes less time and effort. In an industrial setting, the reduction in programming time tends to translate into an increase in productivity and reduction in cost.
- PbD does not require a robotics expert to be performed. This not only eliminates the need to rely on an expert but also increases the accessibility of robots by the general public.
- PbD shows much more promise in transferring human skills to robots. Traditional programming often fails to capture the intricate details of a given task.

[1]Serdar Hakan Arguz is a master's student in the Department of Mechanical Engineering, Izmir Institute of Technology, Turkey serdararguz@iyte.edu.tr

[2]Şeniz Ertuğrul is with the Department of Mechatronics Engineering, Izmir University of Economics, Turkey seniz.ertugrul@ieu.edu.tr

[3]Kerem Altun is with the Department of Mechanical Engineering, Izmir Institute of Technology, Turkey keremaltun@iyte.edu.tr

Robotic assembly is one of the largest areas of robotics that can greatly benefit from these advantages. However, there are still problems to be tackled for a more widespread adoption of PbD in the industry [5]. For the scope of this paper, there are two prominent issues that are addressed by the research in the field.

The first issue is the development and implementation of overarching frameworks that address the learning of the whole assembly process from start to finish [4] [6]. Such frameworks often rely on vision-based systems to decompose a demonstrated assembly operation into a sequence of smaller tasks. For example, a typical assembly operation includes many peg-in-hole tasks [5]. These tasks are then learned using methods such as Gaussian Mixture Models (GMM), Hidden Markov Models (HMM), or Dynamic Movement Primitives (DMP). The proposed frameworks, on the other hand, represent the relationship between the individual tasks and the overall process, so that an appropriate sequence of tasks can be generated when encountered with a novel assembly operation [4] [7].

The second important issue is the PbD's need to provide generalizability, specifically for the tasks that are encountered during assembly operations. Generalization refers to the idea that the objective of a task should still be accomplished, even in novel cases that are not encountered during the demonstrations. For example, a peg-in-hole task is generalized if the peg can still be inserted into the hole successfully, even if the hole is at a different position, or its dimensions are slightly different. To accomplish this, learning methods such as DMP can generate new trajectories for novel hole and peg poses, using the data recorded during the demonstrations [4]. Even though the methods such as DMP can provide some level of generalization capability, there is no guarantee that all the generalizations of a task will be successful. For example, the literature suggests that the success rate for a peg-in-hole task can be as high as 90% [4], or as low as 50% [8]. Trivially, the demonstrations themselves can be replayed with a practically 100% success rate. As the similarity of the task to the one for which the demonstrations are performed decreases, the success rate is also expected to drop.

This study explicates the relationship between the novelty and the success rate of a peg-in-hole task. To do so, demonstrations are performed on a UR5 manipulator and the DMP method is applied to generate trajectories for varying hole positions. The necessary background information on this process is discussed in section II. Section III outlines the methodology used for the experimental evaluation of the success rate. The results of the experiments are then presented in section IV. Finally, section V provides the conclusions with a discussion on the future work that can be undertaken to increase the success rate.

II. THEORY

This section outlines the steps of the PbD process and highlights the aspects that are relevant to this particular study. The PbD process consists of three stages [1]. The first is the acquisition of data from the demonstrations. In this step, a set of variables are recorded as the task is performed by a teacher. The second step is the encoding and representation of this data to obtain a model of the demonstrated task. This is the step at which learning occurs. In the third step, the task is implemented so that the robot can perform it by itself. Additionally, some literature also considers a fourth step: refinement [3] [9]. This last step aims to enhance the adaptiveness and quality of the execution, generally with the aid of other learning methods such as Reinforcement Learning (RL) [3]. Throughout this document, these steps will be referred to as demonstration, representation, implementation, and refinement.

A. Demonstration

The demonstration is the step at which the robot observes the task performed by a human teacher. Decisions about *how* and *what* to observe as a demonstration constitutes the two main design choices relating to this step. An answer to the question of *how to observe* is given by the preferred modality of the demonstration, whereas *what to observe* is answered by the selection of variables to record.

The modality of demonstration describes the means by which the robot observes the demonstration. For example, the use of sensors on a human teacher, teleoperation, and kinesthetic teaching may all be given as examples to different modalities [10]. In this study, the demonstrations are performed using the kinesthetic teaching modality using the free-drive functionality of UR5.

The second important decision about the demonstration is the set of variables to record. This information constitutes what the robot sees as a demonstration. On a low-level, either task space or joint space variables can be recorded. Alternatively, one can record high-level state-action pairs. However, the definitions of those high-level states, such as "holding an object", "moving forward", etc. still need to be defined in terms of low-level variables. This study records joint space trajectories via ROS, as they are published under the \joint_states topic by the UR ROS Driver.

B. Representation

The representation step consists of encoding the acquired data in such a way that a model of the demonstrated task is constructed. Using this model, the robot can not only mimic the demonstration but learn the task from it. That is, it gains the capability to generalize the task for novel situations.

Two main ways in which the representation can be done are often referred to as the high-level and the low-level representations [3]. The high-level representation encodes the demonstrations in a symbolic way, generally as a directed graph between states. The low-level representation, on the other hand, works with the trajectories of the recorded variables. The model is then generated using one of the following three commonly used methods: Hidden Markov Models (HMM), Gaussian Mixture Models (GMM), and Dynamic Movement Primitives (DMP) [1] [3].

HMM and GMM use a statistical approach to create a model from a given set of demonstrations, whereas DMP takes a dynamical systems approach. This study employs the DMP method to learn from the demonstrations.

1) Dynamic Movement Primitives: DMP is a method that uses the equations of a dynamical system to generate trajectories [11]. When doing so, it uses the dynamics of a second-order system due to its well-known convergence and stability properties. The equation for a mass-spring-damper system with the equlibrium position $y = g$ can be arranged in the following way:

$$\ddot{y} = \alpha_y \left(\beta_y \left(y - g \right) - \dot{y} \right) \tag{1}$$

The coefficients α_y and β_y are generally selected in a way that makes the system critically damped, which corresponds to $\alpha_y = 4\beta_y$. This equation could be used to generate trajectories that converge to any desired goal position $y = g$, starting from any initial condition. However, doing so would also fully determine the velocity with which the generated trajectory is followed. To provide an adjustable curve velocity to the solution, a new variable z is defined to be the original curve velocity multiplied by some time constant τ of our own choice.

$$\tau \dot{y} = z \tag{2}$$

Equation (1) can be re-written using this new variable z. By substutiting $\dot{y} = \frac{z}{\tau}$, $\ddot{y} = \frac{\dot{z}}{\tau}$ and multiplying the whole equation by τ^2 , the following equation is obtained.

$$\tau \dot{z} = \alpha_z \left(\beta_z \left(y - g \right) - z \right) \tag{3}$$

Here, $\alpha_z = \tau\alpha_y$ and $\beta_z = \tau\beta_y$. The condition to make the system critically damped is still $\alpha_z = 4\beta_z$. Equations (2) and (3) define a second-order system using a system of two first-order differential equations. Any trajectory that is generated using the dynamics of this system would end up showing the characteristics of a critically damped second-order system response. However, the method requires the capability to express any arbitrary curve so that it can encode the demonstrations. To do so, deviations from the unforced system response are necessary. All the effects that correspond to these deviations can be provided by a forcing term, which is added to the (3) as follows:

$$\tau \dot{z} = \alpha_z \left(\beta_z \left(y - g \right) - z \right) + f \tag{4}$$

To maintain the stability and convergence characteristics of the second-order system, we require the forcing term f to approach zero, as the trajectory approaches the goal. To express this idea, we use a phase variable x, and express the driving function as a function of it. A reasonable choice for the phase variable would be a quantity that represents how much of the task is completed so far. That is, the phase variable is selected to be 1 at the initial condition, and approach to 0 as the trajectory approaches towards the goal. An exponential decay function of the form $x = e^{\frac{-\alpha_x t}{\tau}}$ has this characteristic and can conveniently be used as a phase variable.

$$\tau \dot{x} = -\alpha_x x \tag{5}$$

Finally, the DMP method defines the following forcing term as a function of the phase variable [11].

$$f(x) = \frac{\sum \psi_i w_i}{\sum \psi_i} x(g - y) \tag{6}$$

In this equation, ψ_i are Gaussian kernels given by the expression:

$$\psi_i(x) = exp(-h_i(x - c_i)^2) \tag{7}$$

where c_i are the centers of the Gaussian kernels and h_i are arbitrarily selectable coefficients describing their width.

To find the weights w_i, we formulate a function approximation problem, first by susbtutiting $z = \tau \dot{y}$ and $\dot{z} = \tau \ddot{y}$ into (4) to express the equation only in terms of $y, \dot{y}, \ddot{y}$.

$$\tau^2 \ddot{y} - \alpha_z \left(\beta_z (g - y) - \tau \dot{y}\right) = f \tag{8}$$

Given a demonstration y_{demo} and its derivatives $\dot{y}_{demo}$, $\ddot{y}_{demo}$, the required driving force to follow that demonstration can be solved from (8) as follows:

$$f_{target} = \tau^2 \ddot{y}_{demo} - \alpha_z \left(\beta_z (g - y_{demo}) - \tau \dot{y}_{demo}\right) \tag{9}$$

The weights w_i is to be selected such that the $f(x)$ computed using (6) approximates f_{target}. The solution to this problem can be found using locally-weighted regression, and is given as: [11]

$$w_i = \frac{\mathbf{s^T \Gamma_i f_{target}}}{\mathbf{s^T \Gamma_i s}} \tag{10}$$

where

$$\mathbf{\Gamma_i} = \begin{bmatrix} \psi_i(x_1) & & & 0 \\ & \psi_i(x_2) & & \\ & & \dots & \\ 0 & & & \psi_i(x_N) \end{bmatrix}$$

$$\mathbf{s} = \begin{bmatrix} x_i \\ x_2 \\ \dots \\ x_N \end{bmatrix} (g - y_0) \qquad \mathbf{f_{target}} = \begin{bmatrix} f_1 \\ f_2 \\ \dots \\ f_N \end{bmatrix}$$

where $x_1, x_2, ..., x_N$ denotes the values of the phase variable x at the times corresponding to each sample of the recorded demonstration. Similarly, f_i denotes the f_{target} calculated using the i-th index of $\ddot{y}_{demo}$, $\dot{y}_{demo}$ and y_{demo}.

Equations (2), (4), and (5) define a system of three first-order differential equations. Once the weights are known, this system can be used to generate a trajectory between any novel initial condition and goal. This is the way in which the DMP method is used to generalize a task.

There are two implementational details of DMP that can be discussed in addition to its theory. The first one is the extension of the formulation presented above to multi-degree-of-freedom systems. In the case of a robot, the DMP method can be used simply by applying it separately to individual joint variables [11]. The second issue is the use of multiple demonstrations. There exist several ways to incorporate data coming from multiple demonstrations into this method. For example, some studies first use the GMM to generate a single trajectory that is representative of all the demonstrations and then use DMP to learn it [4]. Alternatively, the DMP formulation can be used as it is, just by using the average f_{target} in the function approximation problem. To do so, the f_{target} computed using the demonstration trajectories $\ddot{y}_{demo}$, $\dot{y}_{demo}$ and y_{demo} have to be added index by index. Considering the fact that the demonstrations do not necessarily have the same time span, some form of stretching might be required at this step. Dynamic Time Warping (DTW) is one of the most commonly used algorithms for this purpose [4]. However, simpler stretching methods also suffice especially if the demonstration durations do not differ largely.

C. *Implementation*

After the representation step provides a model, the robot can start performing the task. To do so often requires the implementation of one or several types of controllers. For a low-level representation approach, these controllers take the form of joint space position and velocity controllers. In this study, the trajectory generated by the DMP method is sent to the UR5 via the Joint Trajectory Action interface, provided by the UR ROS Driver.

D. *Refinement*

The last step of the PbD process can be taken as the refinement of a learned model [9]. This can be done either by using human feedback on the performance of the execution or by the use of other machine learning methods such as RL. This extra step, if incorporated, provides two main advantages to the PbD process. The first advantage is the optimization of the task. It is often the case that the demonstrations are performed in a sub-optimal way by the human teacher, with respect to some desired metric. The refinement process can improve the efficiency and quality of the execution by optimizing that metric. Secondly, the refinement process with human feedback can be used as means to gather more demonstrations only as needed. Doing so reduces the number of demonstrations initially required to start executing the task.

III. METHODOLOGY

This section outlines the specific procedures and equipment used for each of the steps of the PbD process. Throughout this study, a hole of 12 mm diameter is used in conjunction with a peg of 10.2 mm diameter. During the demonstrations, the hole is maintained at a fixed position in the workspace of the robot. The initial conditions for each demonstration are altered without any pattern while keeping the goal position the same. Throughout this document, this position will be referred to as the original hole position. The peg-in-hole task is then demonstrated by a human teacher on a UR5 manipulator using kinesthetic teaching modality by manually inserting the peg into the hole, positioned at the original hole position. Fig. 1 shows the manner in which the demonstrations are performed.

The joint space trajectories corresponding to the demonstrations are recorded into the computer via the UR ROS Driver. The DMP method is applied to learn from these trajectories. The resulting dynamical system is used to generate novel trajectories for varying hole positions. There are a total of 32 hole positions, selected at increasing distances from the original position. They are arranged at 4 distance levels, forming co-centric circles with radii of 30 mm, 60 mm, 90 mm, and 120 mm around the original position. On each circle, there are 8 holes placed with equal spacing of 45°. Fig. 2 shows a diagram of the hole positions.

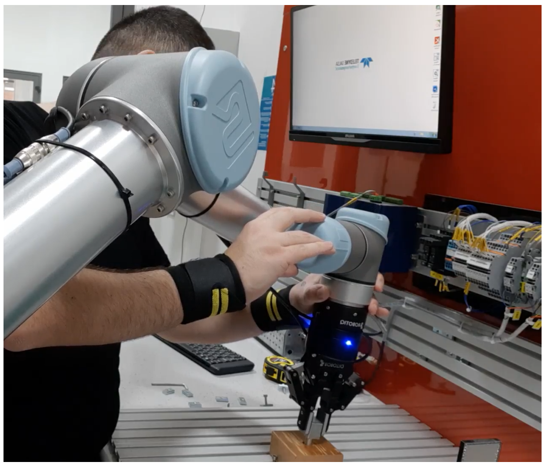

Fig. 1. Demonstrations are performed using a UR5 manipulator

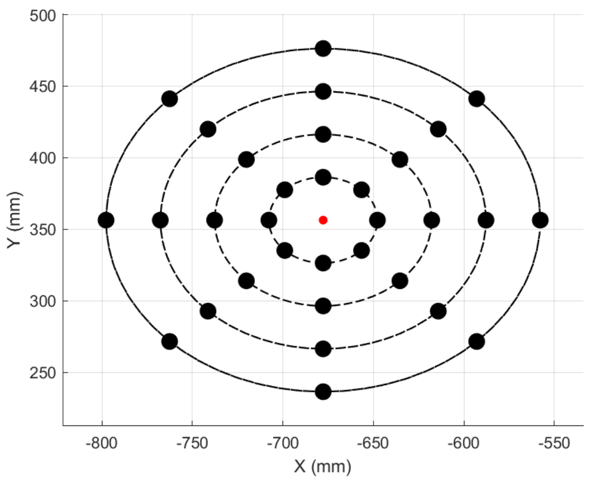

Fig. 2. Hole positions shown as circles, at their corresponding task space coordinates

For each of the 32 trials, the hole is positioned at the designated place. The trajectory generated specifically for that hole position is executed on the robot. In the cases in which the robot cannot insert the peg into the hole, it goes into a protective stop due to the reaction forces it experiences. These cases are taken to be failures. Trials at which the peg is inserted into the hole without a problem are accepted to be successful.

IV. RESULTS

Demonstrations are performed according to the methodology outlined in the previous section. The joint space trajectories associated with these demonstrations are shown in Fig. 3. Additionally, the corresponding task space trajectories are computed and provided in Fig. 4. Two important remarks can be made about these demonstrations.

The first remark is about the final configurations. Even though the task space trajectories all terminate at the fixed hole position indicated by the red circle in Fig. 4, the joint space trajectories do not have exactly the same final values. That indicates that even though the peg is inserted into the same hole in all demonstrations, its final orientation and the depth of penetration into the hole are slightly different at each time.

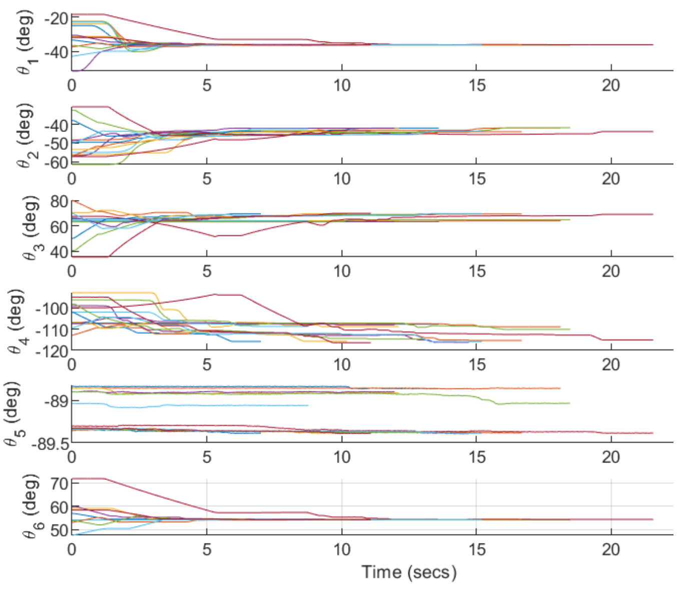

Fig. 3. Demonstration data shown in the joint space

A second remark can be made about the shape of the curves followed by the task space demonstrations. As can be seen in Fig. 4, they include many maneuvers to adjust the orientation of the peg and its alignment with the hole. This results in patterns where the peg approaches the hole with a long and complicated path, instead of following the shortest path. In one aspect, this constitutes one of the ways in which the human demonstrations are imperfect, whereas on the other hand, this is precisely the type of intricate detail that the PbD attempts to capture.

The trajectories recorded during the demonstrations are fed into the DMP method. The resulting dynamical system is

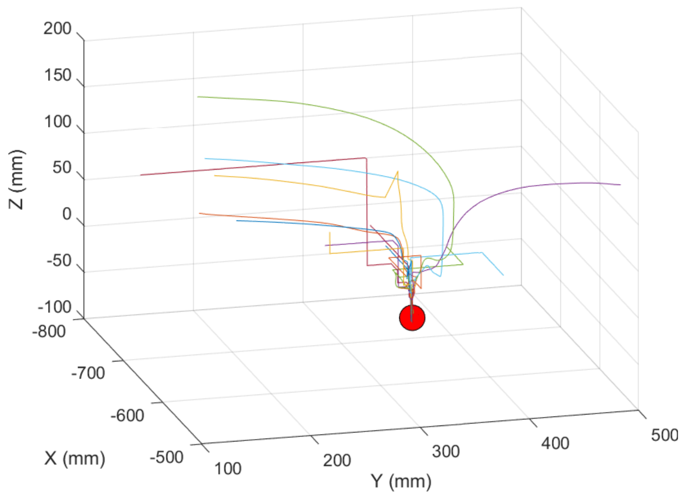

Fig. 4. Demonstration data shown in the task space, computed using the forward kinematics

used to generate 32 trajectories, each with a goal corresponding to a different hole position. For the sake of comparison between them, the initial condition for all these solutions is taken to be at a location directly above the original hole position. Fig. 5 shows a task space plot of the hole positions in black circles and the generated trajectories in red dashed lines.

The holes are placed at their designated locations and the trajectories are executed. The corresponding results are shown in Fig. 6. The hole locations that are highlighted in green are the ones at which the task is successfully executed. Similarly, the hole locations that are highlighted in red indicate the failures. It is worth mentioning that each individual trial is performed several times. Since the hole clearance is well above the repeatability of the UR5 manipulator, the success or failure of the task at a specific hole location remains the same between all repetitions.

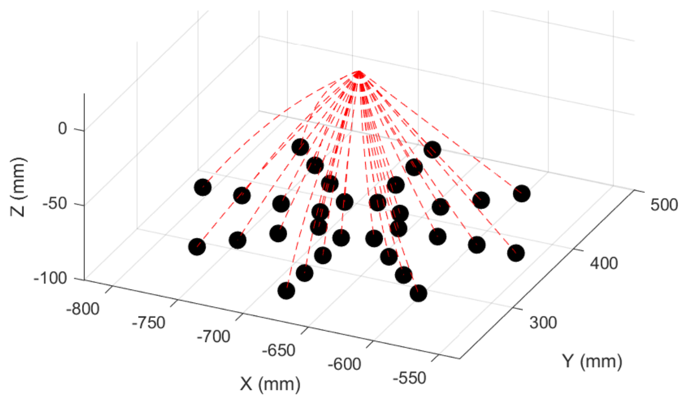

Fig. 5. Hole positions (black circles), and the generated trajectories (red dashed lines)

The hole positions are organized along circles with increasing radii. On each circle, there exist 8 holes, with $45°$ spacing. As can be seen from Fig. 6, the success at any radius level is not uniform. That is, there are some cases where the task fails even performed at the same distance from the original hole position. However, there does not seem to be a systematic influence of the angular position of the hole to the failure. This effect of local failure or success at a specific distance can be explained by the way in which the generated trajectory approaches the hole. Since DMP mimics the geometric characteristics of the demonstrations, the curvature characteristics of the curves shown in Fig. 4 contribute to the success or failure in non-obvious ways. On the other hand, there exists a clear relationship between the radius of the circle on which the hole is positioned, and the success ratio for the trials along that circle. Using the distance from the original hole position as a metric for the novelty of the task, it is found that the success ratio of a peg-in-hole task, learned using the DMP method, decreases as the novelty of the task increases. This relationship is most clearly shown in Fig. 7, where the horizontal axis shows the distance of the hole from the original hole position, and the vertical axis, the success rate of the trials performed at that distance.

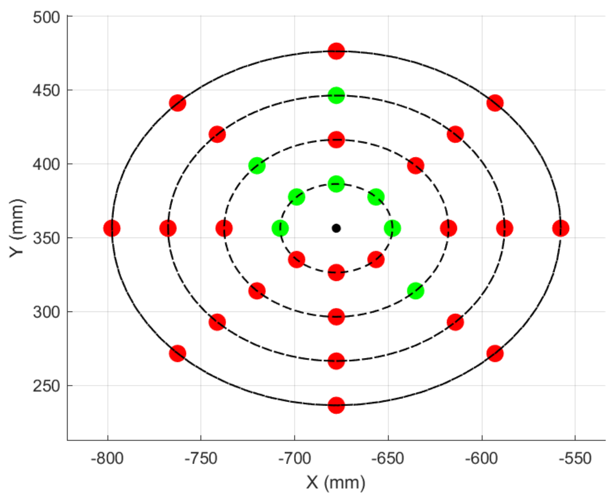

Fig. 6. Holes at which the task was successfull (green) or not (red)

V. CONCLUSIONS

PbD is an approach to robot programming that has many benefits over the traditional method. These benefits, such as human to robot skill transfer, are of great importance to the field of robotic assembly. Concerning the application of PbD methods in robotic assembly, one of the most important aims is the generalization of a learned task. Since a peg-in-hole type of task is commonly accepted to be the most typical example of an assembly task, the generalization performance of a PbD method can be investigated by applying it to novel peg-in-hole tasks. This study, in particular, experimentally evaluates the success of novel peg-in-hole tasks learned from demonstrations using the DMP method.

The novelty of a task is an aspect that is not easy to quantify. However, for a simple peg-in-hole task, the distance of the hole position at which the trial is performed to the position at which the demonstrations were gathered can be

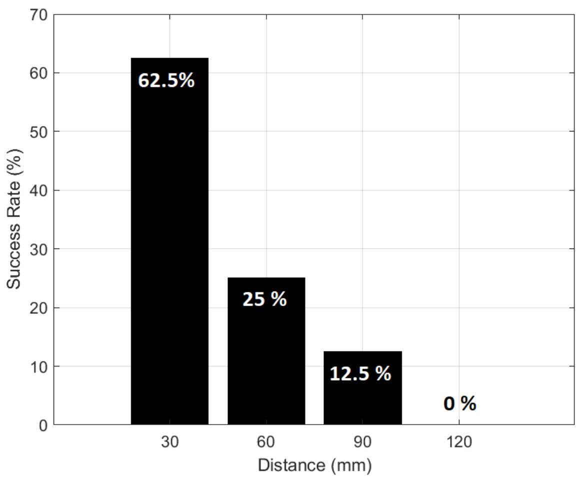

Fig. 7. Success rate vs the distance from the original hole position

taken as a metric for the novelty of the task. Similarly, the success ratio of the trials performed at a specific distance away from the original position can be used as a metric for the generalization performance.

To obtain an experimental relation between these metrics, the process of PbD is applied in the following fashion. The demonstration step is carried out by recording the joint space trajectories encountered during the execution of a peg-in-hole task by a human teacher. In the representation step, recorded demonstrations are used to learn the task using the DMP method. The dynamical system defined by the DMP method is then used to generate novel trajectories for 32 different hole positions. These hole positions lie along 4 co-centric circles with increasing radii, so that the effect of distance can be tested. The generated trajectories are executed on a UR5 manipulator. Cases at which the robot entered into a protective stop due to high reaction forces are taken to be failures. The other cases in which the peg is successfully inserted into the hole are accepted to be successful. The success at each hole location and the success ratio at different distances from the original hole position are evaluated.

The results show that the generalization performance of a peg-in-hole task, performed using the DMP method, decreases as the novelty of the task increases. The success ratio obtained at the holes 30 mm away from the original position is 62.5%, which agrees with the results previously found in the literature [8] [4]. As the distance increases, however, the success ratio rapidly drops. For the hole positions 120 mm away from the original position, all trials result in a failure.

It is worth mentioning that even though these results clearly show the relationship between the novelty and the generalization performance, they are still dependent on the particularities of the experiment. The success ratio for each distance level could have been increased by reducing the required depth of penetration into the hole or increasing the clearance of the hole. However, it is expected that the decreasing relationship between the success ratio and the distance would remain the same, even though the particular success ratios can change. A similar thing can be said about the quality of the demonstrations. The quality of the demonstrations might affect exactly which hole positions fail or not. It might be possible to find another human teacher for which some of the failed trials would result in success. Nevertheless, the relationship shown in this study is still expected to hold. The specific effects of the characteristics of demonstrations on the failure and success at particular positions is a topic for future studies. For example, it can be speculated that if most of the demonstrations are performed by approaching the hole position from left, the method may show more success in generalizing the task successfully for novel hole positions along that direction.

The future work on the topic includes strategies using which the success rate can be increased for any hole location. The most prominent category of such strategies is the algorithms that allow the robot to maneuver during the run time. Importantly, force feedback can be used as a way for the robot to understand if it is approaching the hole correctly or not. Novel algorithms that constitute how such a task can be performed are an important topic for future research.

References

[1] S. Ambhore, "A comprehensive study on robot learning from demonstration," in *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, pp. 291–299, IEEE, 2020.

[2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[3] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Survey: Robot programming by demonstration," *Handbook of robotics*, vol. 59, no. BOOK_CHAP, 2008.

[4] G. Ding, Y. Liu, X. Zang, X. Zhang, G. Liu, and J. Zhao, "A task-learning strategy for robotic assembly tasks from human demonstrations," *Sensors*, vol. 20, no. 19, p. 5505, 2020.

[5] Z. Zhu and H. Hu, "Robot learning from demonstration in robotic assembly: A survey," *Robotics*, vol. 7, no. 2, p. 17, 2018.

[6] Y. Wang, R. Xiong, L. Shen, K. Sun, J. Zhang, and L. Qi, "Towards learning from demonstration system for parts assembly: A graph based representation for knowledge," in *The 4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent*, pp. 174–179, IEEE, 2014.

[7] Z. Zhu, *Robot Learning Assembly Tasks from Human Demonstrations*. PhD thesis, University of Essex, 2020.

[8] D. A. Duque, F. A. Prieto, and J. G. Hoyos, "Trajectory generation for robotic assembly operations using learning by demonstration," *Robotics and Computer-Integrated Manufacturing*, vol. 57, pp. 292–302, 2019.

[9] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[10] B. Fang, S. Jia, D. Guo, M. Xu, S. Wen, and F. Sun, "Survey of imitation learning for robotic manipulation," *International Journal of Intelligent Robotics and Applications*, vol. 3, no. 4, pp. 362–369, 2019.

[11] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.